Accessing the Data Efficiently using Prediction of Dynamic Data Algorithm

Annamalai R
Assistant Professor
Jeppiaar Institute of Technology
Sriperumbudur, Chennai

Srikanth J
Assistant Professor
Jeppiaar Institute of Technology
Sriperumbudur, Chennai

M Prakash, Ph.D
Associate Professor
Jeppiaar Institute of Technology
Sriperumbudur, Chennai.

ABSTRACT
The Internet carries an extensive range of information resources and services throughout the world. Without Internet finding a particular thing, or get details of those things is not possible. In this project the system focused on accessing the data efficiently through offline or online. In addition of this project the proposed system developed an algorithm to differentiate static and dynamic data. It is called prediction of dynamic data algorithm. This project aims to quick access of data and other information. The System Proposals implement this project in android application. With the help of Sqlite Database we store and retrieve the data in the Mobile Application. Sqlite Database is a relational database management system; it is self-contained and most widely deployed Sql database engine in the world. In this project there is no need of internet connectivity if the data is stored in local database. Charge in the device consumes less when compared to the usage of existing system. There is no need for user has to login in the mobile application, it is optional only. It provides Security from the malware content. Thus the data will be secure and protect from viruses.

General Terms
Websites, Mobile Application, Retrieving Data from website, YQL Console

Keywords
Offline access data, Data Retrieval, Static Data, Dynamic Data, Search View.

1. INTRODUCTION
Software for business applications, whether it is intended to perform a single task or it is Intended for use as a company-wide, integrated system, should be tailored to fit the Company's unique needs and goals. In the simplest terms, the software should be capable; by performing all the functions necessary to perform a task efficiently manuscripts must be in English. These guidelines include complete descriptions of the fonts, spacing, and related information.

Android is a mobile operating system (OS) based on the Linux kernel and currently developed by Google. With a user interface based on direct manipulation, Android is designed primarily for touch screen mobile devices such as smart phones and tablet computers, with specialized user interfaces for televisions (Android TV), cars (Android Auto), and wrist watches. The OS uses touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects, and a virtual keyboard. Despite being primarily designed for touch screen input, it also has been used in game consoles, digital cameras, and other electronics.

In this system proposal using touch screen input on the android device, most frequently searched data in the internet can be viewed in offline. Most frequently searched data can be Wikipedia content, meaning for the words, nearby services using gps etc.

2. RELATED WORK
In Existing System the Internet Connectivity is mandatory for accessing information. Sometimes there is a lot of wrong information on the internet, anyone can post anything, and much of it is garbage. There are predators that hang out on the internet waiting to get unsuspecting people in dangerous situations. Hackers can create viruses that can get into your personal computer and ruin valuable data. Charge in the Device consumes more due to the internet connection.[2]

In proposed system provides accessing public information through offline. It does not need Internet Connectivity if the data is in local database. The Information in the offline app is accurate and verified information only updated. It provides Security from the malware content. Thus the data will be secure and protect from viruses. Charge in the Device consumes less compared to using in the Internet Connection. There is no need for user has to login in the mobile application, it is optional only.

3. SYSTEM MODEL
The System Model consists of Web Application Development, Server and Database, Admin control over the mobile application from server side, Google server. At the Initial stage, data should be collected and stored in local database. Nearby searvice information are get from the social Websites like Justdial.com, yellowpages.com. Through YQL console these data are get and stored into the database.[3]

After getting the data, it was arranged and put it into the mobile application(i.e hosted into the google server).

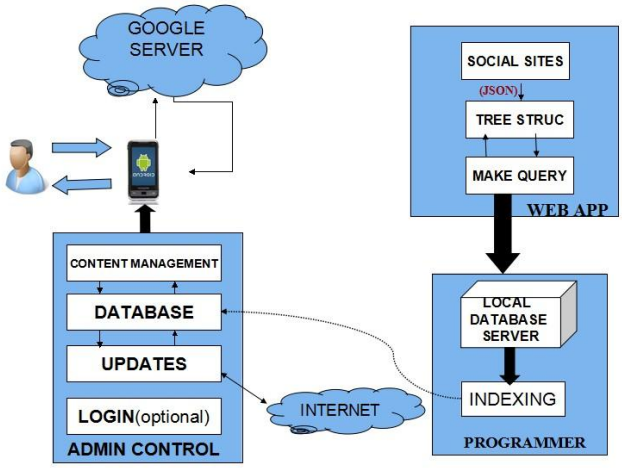

Fig 1: System Model

4. IMPLEMENTATION

The Information Stored in the mobile application should be correct and should not include any duplicate data. In this Part, the Social Sites will be any, according to the particular area of the downloading application. It was Chosen depend on the most searchable sites and Common Websites like justdial.com. In this module, the analyzed websites are making into Tree Structure i.e. data as nodes using the XML and JSON[5]. Using YQL Console the elements in the Website are converted into Parent Nodes, the data are in the Leaf Nodes.

The data in the leaf node can be retrieved as a information using Php, and store it into the Mysql Database. Then the data can be arranged in the required manner. Then, the data are pushed into the mobile application. According to the user needs it will fetch the data from database. In addition, the mobile application predicts the user input whether the input is static data or the dynamic data. Static data are the data which stored in the database, where dynamic data is the data available in the internet i.e. it is an online requirement for that input. If it is a dynamic data then the result is fetch from the internet. The user input in the android application is different from one another. The input looks like, the input enter in an google search engine. Thus, for effective autocomplete text and retrieval data a PODD(Prediction of Dynamic Data Algorithm) is implemented.

Fig. 2 shows the Execution time for store the data from websites to the local database using YQL Console. Each row(data) takes approximately one or less than a second to store into the Mysql database.

The query used in the YQL console is “select * from html where url=’ ‘ “, inside the single quotes website url is placed e.g. www.justdial.com. Then using Javascript Object Notation (JSON) data in the tree structure is converted into Mysql database table. With the help of Php language and Mysql database the data can be stored.

5. PREDICTION OF DYNAMIC DATA ALGORITHM

The System uses the concept of Spelling Corrector Technique. It works on the basis of Probability Theory. Give a word, we are trying to choose the most likely spelling correction for that word (the “correction” may be the original word itself). There is no way to know for sure (for example, should “lates” be corrected to “late” or “latest”?), which suggests we use probabilities.

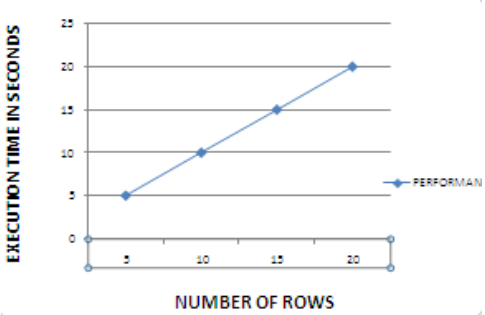

Fig 2: Number of Rows per Execution Time in seconds

The System will say that we are trying to find the correction c, out of all possible corrections, that maximizes the probability of c given the original word w: Consider the misspelled word w=”thew” and the two candidate corrections c=”the” and c=”thaw”. Which has higher P(c|w). Well, “thaw” seems good because the only change is “a” to “e”, which is a small change.

On the other hand, “the” seems good because “the” is a very common word, and perhaps the typist’s finger slipped off the “e” onto the “w”. The point is that to estimate P(c|w) we have to consider both the probability of c and the probability of the change from c to w anyway, so it is cleaner to formally separate the two factors.

The System will read a big text file, big.txt, which consists of about a million words. The file is a concatenation of several

public domain books from Project Gutenberg and lists of most frequent words from Wiki dictionary and the British National Corpus.

We then extract the individual words from the file. Next we train a probability model, which is a fancy way of saying we count how many times each word occurs, using the function train. From this we get the highest probability of the word. With this technique we predict the dynamic data compared with the offline content.

Now let's look at the problem of enumerating the possible corrections c of a given word w. It is common to talk of the edit distance between two words: the number of edits it would take to turn one into the other.

An edit can be a deletion (remove one letter), a transposition (swap adjacent letters), an alteration (change one letter to another) or an insertion (add a letter). This can be a big set. For a word of length n, theorem will be n deletions, n-1 transpositions, 26n alterations, and 26(n+1) insertions, for a total of 54n+25 (of which a few are typically duplicates). For example, len(edits 1('something')) – that is, the number of elements in the result of edits1('something') – is 494.

The literature on spelling correction claims that 80 to 95% of spelling errors are an edit distance of 1 from the target. Perhaps the examples found are harder than typical errors. Anyway, it was not good enough, so will need to consider edit distance 2. That's easy: just apply edits1 to all the results of edits1: def edits2 (word): return set (e2 for e1 in edits1(word) for e2 in edits1(e1): This is easy to write, but starting to get into some serious computation: len(edits2('something')) is 114,324. However, we do get good coverage: of the 270 test cases, only 3 have an edit distance greater than 2. That is, edits2 will cover 98.9% of the cases, that's good enough. Since it was not going beyond edit distance 2. Thus with the help of spelling corrector technique the static data is retrieved from the Sqlite database (stored in the android Application) using Indexing Technique.

App Distribution and Installation

The Mobile Application is put in live in the cloud for the user access. The Installation process is very simple and it doesn't need any registration process. (Optional).

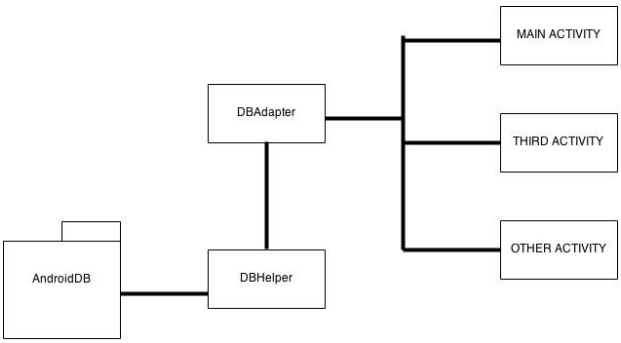

Fig 3: Interface between the Database Adapter and Activity

Updates and Security Evaluation Metrics

Update process are quite simple in this mobile application. It doesn't update the entire application. Only the new values (data) are updated in the application.

6. EXPERIMENTAL RESULTS

Testing is a broad subject, with various methodologies combining various approaches, but it embodies attempts to ensure that the user's experience is satisfactory: intuitive, reliable, responsive, secure, etc. Practices such as Test-Driven Development and Behavior -Driven Development are popular at the time of writing, using means such as Unit Testing and Integration Testing, but wider testing areas include checking that the user can perceive how to achieve his/her goals without frustration or cultural misunderstandings through to checking that the server against which an application runs can sustain the expected average throughput, gracefully handles peak load, is robust against attacks, and so on.

The android.jar on the development system (against which applications are built) throws exceptions in response to attempts to execute its methods.

Android declares several of these exception-throwing methods as final, rendering them impossible to include in tested code directly.

Android makes use of static methods, again impossible to test directly.

The official Android documentation tends to focus on tests based on the Android Test Case and its subclasses, with commentators often calling them 'unit tests'. These tests are emphatically not unit tests: since they integrate with the Android system, they are at least integration tests, but while they are too embedded and slow to execute as unit tests, they are too tightly coupled to the code itself to be system testing. Because they run on a phone, the compile/execute cycle of the unit test becomes a compile/package/install/execute cycle, but they run against the real implementation and thus don't suffer from the danger of implementation inaccuracy unit test stubs and mocks present.

7. CONCLUSION & FUTURE WORK

In this paper the system store and retrieve the data in the Mobile Application in Android Environment using Sqlite Database. It provides necessary information for the users. The data stored in the mobile Application was verified and checked simultaneously. The data includes the Hospitals, Restaurants, Bank, ATM's etc. It also provides the contact information and the address of the respected hospitals, banks, etc. The data can be search inside the data i.e the system can search the place or address as the keyword. Admin can also monitor the searchable data by the users. It does not show any unwanted data or advertisements.

Future enhancement of this project will include all stores and information in the City. It will also include the maps and Wikipedia content as offline. Analysis between the static and dynamic data. To compare and analyze other related items with the stored data. Storing the searched data into the table for the future autocomplete option. The Future Work also includes suggestions for the users convenient searching relevant topics. The performance also is analyzed.

8. REFERENCES

[1] Yahoo! Developer Network. Yahoo query language (YQL) guides 2011. http://developer.yahoo.com/yql/guide/yql_set.pdf.

[2] Moira C. Norrie and Beat Signer. Information server for highly-connected cross-media publishing. Information Systems, 30(7):526-542, November 2003.

[3] Andreas A. Veglis. Modeling cross media publishing. In The Third International Conference on Internet and Web Applications and Services, pages 267 – 272, 2008.

[4] Chen, J. Wu, H. Jian, H. Deng, and Z. Wu 2013. Instant recommendation for web services composition. IEEE Trans. Services Comput. 99,1.

[5] Reto Meier (2009) "Professional Android Application Development" – Wiley Publishing Inc.

[6] Satya Komatineni(2009) "Pro Android" – Apress Publication.